5-31-2015

Crowdsourcing Computer Security Attack Trees

Matthew Tentilucci
Rochester Institute of Technology

Nick Roberts
Rochester Institute of Technology

Shreshth Kandari
Rochester Institute of Technology

Daryl Johnson
Rochester Institute of Technology

Bill Stackpole
Rochester Institute of Technology

See next page for additional authors

Authors
Matthew Tentilucci, Nick Roberts, Shreshth Kandari, Daryl Johnson, Bill Stackpole, and George Markowsky

Crowdsourcing Computer Security Attack Trees

Matthew Tentilucci[1], Nick Roberts[1], Shreshth Kandari[1], Daryl Johnson[1]
Dan Bogaard[1], Bill Stackpole[1], and George Markowsky[2]
[1]Department of Computing Security, Rochester Institute of Technology, Rochester, NY
[2]School of Computing and Information Science, University of Maine, Orono, ME

***Abstract*—This paper describes an open-source project called RATCHET whose goal is to create software that can be used by large groups of people to construct attack trees. The value of an attack tree increases when the attack tree explores more scenarios. Crowdsourcing an attack tree reduces the possibility that some options might be overlooked. RATCHET has been tested in classroom settings with positive results. This paper gives an overview of RATCHET and describes some of the features that we plan to add.**

***Keywords*—crowdsourcing, attack tree, security, attack surface**

I. Introduction

Attack tree analysis, as described by Brue Schneier in the book *Secrets and Lies* [1], is the process of analyzing how systems fail. Attack trees allow users to understand possible threats against systems, visualize those threats and assign various metrics to determine which threats are most likely to occur. Fault tree analysis (FTA), similar to attack tree analysis, has been used since the early 1960s to perform safety and reliability evaluations in high-hazard industries including originally the U.S. Air Force Ballistic Systems Division [2]. While attack tree analysis and fault tree analysis are used in information technology and industrial engineering respectively, the two methods have much in common.

Crowdsourcing is the process of harnessing the contributions of a community of online users to supply services, concepts, or content. Participants bring a rich background of experience, perspective, and expertise to the problem.

Attack tree analysis is difficult to perform well as the volume of detail that must be collected and collated to build a useful tree is large and continually growing. This project combines the ability of attack tree analysis to describe computer security threats with the power of crowdsourcing to create all-encompassing, open source, and community created and maintained computer security attack trees.

There are several incentives to combine attack tree analysis and crowdsourcing. These include visualization of security threats, improved security, and quantification of security efforts. Benefits to the computer security community can only be realized if attack trees are sufficiently comprehensive to allow them to address credible threats to a computer system. These may include vulnerabilities for software and services, multi-step attacks, social engineering, physical security, network device security, etc. The number of attack vectors is too large for a typical organization to understand all of them. Crowdsourcing can address this problem by distributing the effort required to build a complete tree. Collaboratively built attack trees can be more complete and accurate, increasing potential benefits to computing security.

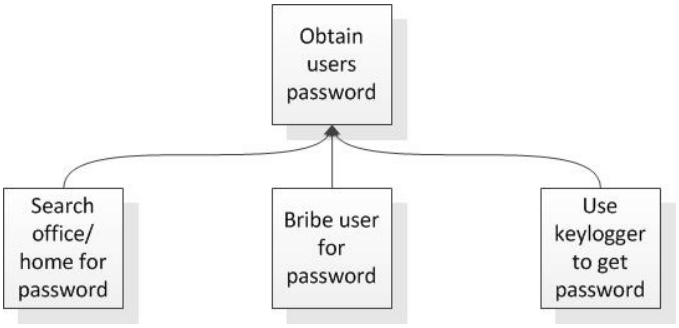

Fig. 1. Starting the Attack Tree

Over the past year a team of faculty and students have implemented a web-based system to allow an online community of users to create attack trees viewable to the general public. The online community has the ability to promote better ideas, voting down the ones perceived as less valuable. RATCHET can found at http://ratchet.csec.rit.edu/. RATCHET permits people to visualize attack trees, share branches, and vote on relevant information. RATCHET also provides a node by node description.

RATCHET combines attack tree analysis with the power of crowdsourcing to create an all encompassing, open source, community created and maintained attack tree system. There are a number of reasons to combine attack tree analysis and crowdsourcing. These include visualization of security threats, improved security, and quantification of security efforts. The benefits to the computer security community afforded by attack trees can only be realized to their fullest if the attack trees created address the most relevant security threats to a system. Given the number of possible attacks, attempting to generate attack trees for all possible attacks is a task too large for small to medium sized organizations. We believe that attack trees created by crowdsourcing will be more complete and will increase the likelihood that major benefits will be realized.

II. Related Work

References [3]–[5] illustrate the advantages of using attack tree analysis and fault tree analysis to model security threats. Zhang et al. [6] show how to supplement fault tree analysis models by adding privilege escalation metrics into the models. Edge et al. [7] introduce the idea of expanding the attack tree concept into a protection tree. They first create an attack tree, calculate the appropriate metrics, and then they create a

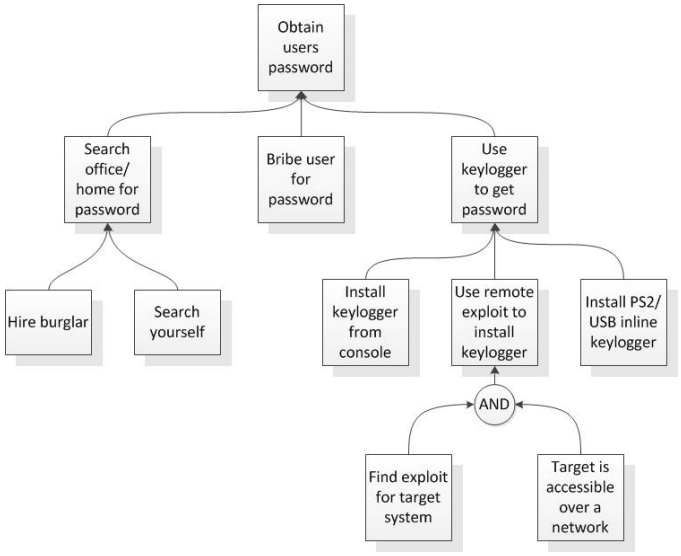

Fig. 2. The Second Attack Tree

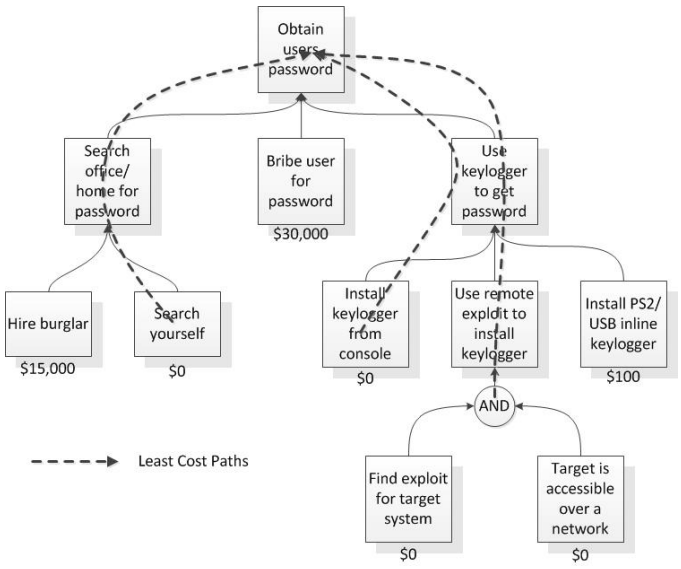

Fig. 3. The Second Attack Tree With Cost Information

protection tree to help planners allocate resources to defend against specific attacks. Roy et al. [10] take the idea of protection trees a step further, proposing attack countermeasure trees where qualitative metrics, defense mechanisms, and probabilistic analysis can be applied to nodes directly within the tree. Bauer [8] discusses scenarios that can be turned into attack trees.

Of special interest is an open-source project called Seamonster [9]. Seamonster is a program designed to produce stand alone attack trees. We will have more to say about Seamonster later in this paper.

III. Project Goals

The goals for this project include: 1) Exploring attack tree analysis and its use in quantifying computer security vulnerabilities and efforts; 2) Creating an attack tree platform that would be community created, maintained, and utilized (e.g. crowdsourced); 3) Implementing features that will facilitate the growth of a new online community focused on the creation of computer security themed attack trees.

Our goal is to have people and organizations use RATCHET to build attack trees specific to their infrastructure and needs, and to share them with others. We expect that the attack trees produced and shared in this manner will greatly benefit the entire Information Technology (IT) community, There currently exists no definitive information source or tool for computer security engineers to use for guidance when adding new devices or services to a network. With attack tree analysis, determining points of failure or faults in current network configurations when introducing new hosts or services into an infrastructure would be more easily performed.

It is also possible that a cost and likelihood could be associated with each attack or vulnerability. With this information, an IT shop might apply a score to their computer security efforts. Such a score could be used as a metric against which to judge whether one is improving one's defenses. Metrics might be used to justify the cost of new devices or services, or to validate the need for a configuration change. Utilizing an attack tree can provide computer security professionals with access to measurable data and help them evaluate the value of adding a new security system or software patch.

IV. Original Vision

A basic attack tree would be focused around a central root node or objective. Such an objective might be to *obtain a users password*. Branching off of a root node, one could create multiple methods to obtain customer data. Fig. 1 shows an example of how one might start an attack tree. The next step might explore what methods one would use to search for a written copy of the password, bribe the user for his or her password, or utlize a keylogger to steal the password. At this stage, a new community of researchers, IT professionals, pentesters, or security minded individuals become a vital part of the process. For example, one individual might know how to burglarize an office, but not how to exploit a system and install a keylogger. Working together, a group is more likely to create a detailed attack tree using different methods of reaching the same goal than would the individuals working alone. An example is shown in Fig. 2.

By using attack trees, organizations can determine where to focus their efforts in order to minimize potential threats. Attack trees that include associated costs, such as the attack tree in Fig. 3, are extremely useful. The attack tree in Fig. 3 can help determine which attacks might be preferred by an attacker who would carry out a low-cost attack.

Price-points may be one method to analyze the value of a given problem, but the cost between organizations might vary based on any number of unrelated factors. Some organizations are less cost-conscious than others. This realization led to a concept that information might need to be assigned different levels of visibility with some of it being global and other items visible only to a particular user. Different attack methods might be displayed with orientation attributes related to different participants-user-oriented vs server-oriented for example. Some attributes that might be of use in analysing cyber attacks

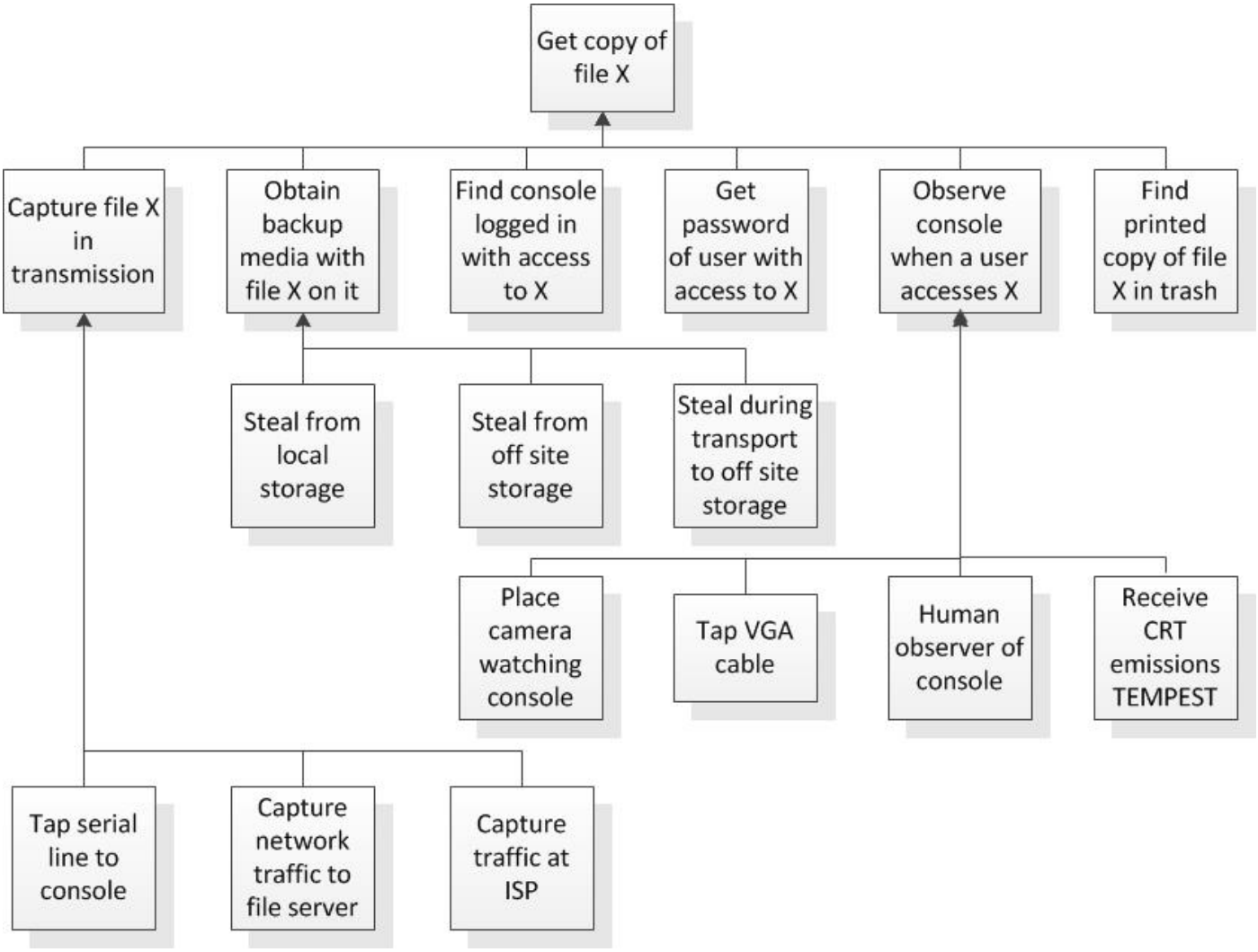

Fig. 4. A More Elaborate Attack Tree

are presented in Table 1. A more elaborate attack tree is shown in Fig. 4.

V. RATCHET

RATCHET is a prototype of a system that can leverage crowdsourcing for the development of attack trees. Web applications provide a very convenient way to reach a lot of people. If properly designed they require only access to a browser which most computer users have. Fig. 5 shows the home page of RATCHET.

The researchers and students involved in this project all collaborated in gathering ideas for what the system should accomplish, how it might be used and what issues it might face. The application was designed for a collaborative audience of users who would have sufficient knowledge and the ability to help the system grow. The system can handle both simple attack trees, e.g., a password attack, and complex attack trees, e.g., attacking an operating system.

RATCHET has the ability to allow users to build a new tree, node by node. Every node in the tree has an area for comments and the capability of allowing users to vote in favor or against it. Any node of any tree can be duplicated and attached to any other node either within the same tree or in completely

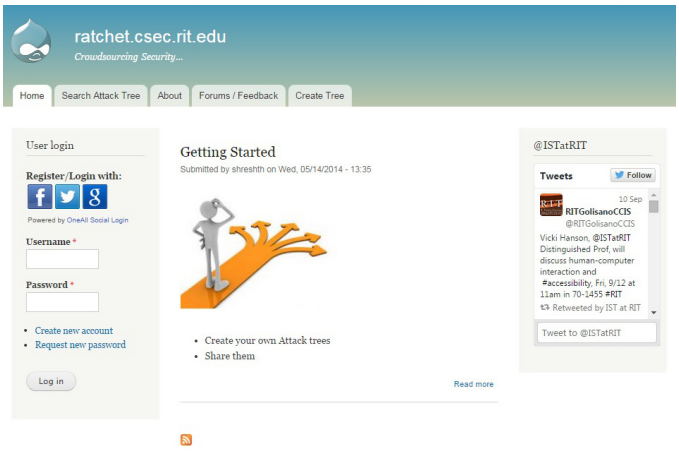

Fig. 5. RATCHET Homepage

TABLE I. EXAMPLE ATTRIBUTES FOR ATTACK TREE ANALYSIS

Attribute	Scale
Cost	Dollars
Difficulty	Easy vs Challenging
Physical Presence Required	Intrusive vs Non-Intrusive
Probability	Possible vs Impossible
Special Equipment (Resources) Required	Availability & Traceability
Risk of Detection	Evidence Left Behind or Stealthiness
Skill Level Required	Novice or Expert

different trees. Each attack tree is displayed visually using Scalable Vector Graphics (SVG - an HTML5 technology that natively allows for dynamically drawing within a web page [11]). The users have the ability to zoom, pan or select any of the created nodes, view the comments and add their own ideas, vote on its relevance, or add a new child node of their own.

We hope to make many improvements to RATCHET based off feedback we have received thus far. Some suggestions we have been thinking about include the ability to work offline, limiting specific trees or nodes to a subset of logged in users and the addition of analysis tools. It would also be good for RATCHET to allow batch uploading of entire trees. This might be done by supplying the user with a JSON (JavaScript Object Notation) format and once the correctly formatted file is uploaded, the system would parse the file and install the tree.

VI. CLASSROOM FEEDBACK

During the Spring semester of 2014, Professors Stackpole and Johnson taught a class titled Penetration Testing Methodologies where they hosted an exercise of the RATCHET system. The goal of this class is to provide students with a realistic

Fig. 6. A Simple Attack Tree in RATCHET

experience with tools, techniques, and goals that face a typical penetration tester or ethical hacker. Students are introduced to the offensive side with items such as Open Source Intelligence (OSINT), scanning, penetration testing framework and collaboration tools, stepping stones, and password cracking. On the defensive side, items such as firewalls, Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), antivirus, software updates, and the like are introduced.

Attack trees were presented as a tool for organizing a penetration testing exercise. The concept of attack trees was raised as the topic for an initial discussion with students. A short training session on how to use RATCHET was presented. Students were broken into four teams with six students per team. Each team was instructed to address two goals: one goal specified the target as an operating system, and the other addressed the exploitation of an application. Each team was instructed to perform a pre-event survey capturing their preconceived notions and opinions on the use and value of attack trees. The class was given a half hour introduction and RATCHET system and its use. Each team was to conceptualize an attack tree to reach their stated goals and to input their attack tree elements into the RATCHET system.

Each team was tasked with reviewing the attack trees created by two other teams, adding content where appropriate, and reviewing the work submitted by their peer teams. The student teams were tasked with using their Attack Trees in performing an attack on a specific target in the Security Lab. Finally the students were asked to provide feedback on the experience of designing their own tree specific to the RATCHET implementation through another survey instrument. This exercise spanned approximately 3 weeks of the course.

From this exercise, several changes were made to the RATCHET system mainly revolving around the interface design. The ability to change the focus of the tree and how to indicate the node on the screen that commands or instructions apply to was enhanced. It was noticed that students did not immediately build or create the attack tree in RATCHET, they built it first on a whiteboard then transposed it to RATCHET. The feedback from students indicates that "In an unfamiliar environment that which is familiar is attractive."

Several issue arose from this exercise that were addressed such as trouble with creating trees, adding new nodes, and voting. While these issues did not stop the exercise they did limit the amount of work that the students could accomplish. As such the exercise was helpful for the developers and enlightening for the students, but the results were of mixed value and will not be included in this paper. Professors Johnson and Stackpole plan to use this exercise in future classes.

In Fall 2014, Professor Markowsky taught a similar course in cybersecurity at the University of Maine that included attack trees as one of the topics studied. Students were encouraged to draw attack trees manually, to use Seamonster [9] and to use RATCHET. Informal feedback was solicited. As might be expected, the manual approach was the easiest approach to start with. It was clear that there were advantages to generating attack trees on the computer since they generally looked better and were easier to share. Seamonster was easy to use, but did support crowdsourcing. Some people thought RATCHET was more complicated to use than Seamonster, but just about

everyone agreed that RATCHET produced the best looking trees. Many students did not like the fact that they were unable to delete nodes once they were created.

In the original design of RATCHET it was decided that users could not delete nodes and that voting would be the way nodes were promoted or demoted. We did not want some user to just delete a tree that many other people had worked on. It was clear that in creating attack trees there would be occasional false steps that users wanted to remove from the tree and they became frustrated by not being able to do so. We plan to explore ways to enable users to delete nodes in limited situations This might take the form of enabling deletion just of nodes created by the user within a single editing session or allowing users to delete nodes that they created but have not been used by anyone working on the attack tree.

VII. FutureWork

We have discussed some of the improvements we hope to implement in RATCHET. In addition to the items already mentioned, there are some other tasks that we would like to complete. First, we would like to construct a library of commonly needed sub-trees that could be copied into a tree under construction to speed up development. Second, we would like to augment the current attribute feature to handle richer structured characteristics such as arrays, tables, or documents. Third, we would like to augment the voting feature to allow voting on the attributes individually.

It addition, we would like to add functions to RATCHET that can perform various analyses of attack trees such as critical path or least cost analysis. Finally, we hope to develop an application program interface (API) to support the development of external tools to utilize RATCHET.

VIII. Conclusions

Compared to other well established field of study such as engineering and chemistry, computing security is still in its infancy. As in the beginning of those other fields, methods, language, and measurements had to be discovered, developed, and generally accepted. The ability to record, exchange, and compare the security state of an environment or situation is necessary in order for the field to progress beyond an art. We believe that attack trees can be used to document, describe, and measure complex, multivariable, and situationally sensitive environments such as the ones found in computing security.

We hope that the impact of attack tree software on the field on computing security will be similar to the impact of electronic spreadsheet programs such as Lotus 1-2-3 or VisiCalc [12]. Just as spreadsheets permit business people to easily do "What if?" scenarios, attack trees can provide cybersecurity people the opportunity to experiment with "What if?" scenarios. Examples of the sort of questions people could ask are: "What if I replaced one firewall product with another?", "What if the Exchange mail server is replaced by the Exim mail server?", "What if remote management is enabled in a particular networking device?". In general, these and other scenarios could be explored quickly and cheaply.

The research and development efforts so far have demonstrated that attack trees can be constructed through the work of a community. The ability of attack trees to record and communicate the attack surface of an operating system, service, and situation has been demonstrated. It has also been demonstrated that attack trees can be shared and assembled to build large and complete attack trees. We look forward to continuing the development of RATCHET and we welcome feedback from the IT community.

References

[1] B. Schneier, *Secrets and lies: digital security in a networked world: with new information about post-9/11 securit*. Indianapolis, Ind.: Wiley, 2004.

[2] C. Ericson, "Fault Tree Analysis - History" *Proc. 17th Int. Syst. Saftey Conf.*, 1999.

[3] P. J. Brooke and R. F. Paige, "Fault trees for security system design and analysis," *Comput. Secur.*, vol. 22, no. 3, pp. 256-264, 2003.

[4] J. B. Odubiyi and C. W. OBrien, "Information security attack tree modeling," *Pract. Exp. Approaches Inf. Secur. Educ.*, pp. 29-37, 2006.

[5] J. L. Bayuk, CISA, and CISM, Stepping Through the InfoSec Program, 1st edition. Rolling Meadows, IL: Isaca, 2007.

[6] T. Zhang, M. Hu, X. Yun, and Y. Zhang, "Computer vulnerability evaluation using fault tree analysis," *Information Security Practice and Experience*, Springer, 2005, pp. 302-313.

[7] K. S. Edge, G. C. Dalton, R. A. Raines, and R. F. Mills, "Using attack and protection trees to analyze threats and defenses to homeland security," *Military Communications Conference*, 2006. MILCOM 2006. IEEE, 2006, pp. 1-7.

[8] M. Bauer, Linux Server Security. OReilly Media, Inc., 2005.

[9] Seamonster - Security Modeling Software, Sourceforge Project, http://sourceforge.net/projects/seamonster/.

[10] A. Roy, D. S. Kim, and K. S. Trivedi, "Cyber security analysis using attack countermeasure trees," *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, 2010, article no. 28, pp. 28-1-28.4.

[11] "W3C." SVG Working Group. N.p., n.d. Web. 30 Oct. 2014.

[12] P. Cunningham and F. Frschl, Electronic Business Revolution: Opportunities and Challenges in the 21st Century. Springer Science & Business Media, 1999.